Information Sciences 678 (2024) 120858

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Adaptive three-way KNN classifier using density-based granular balls

Jie Yang [a,b], Juncheng Kuang [a], Guoyin Wang [a], Qinghua Zhang [a], Yanmin Liu [b], Qun Liu [a], Deyou Xia [a], Shuai Li [a], Xiaoqi Wang [a], Di Wu [a,c,*]

[a] Chongqing Key Laboratory of Computational Intelligence, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
[b] School of information engineering, Zunyi Normal University, Zunyi 563002, China
[c] College of Computer and Information Science, Southwest University, Chongqing 400715, China

ARTICLE INFO

Keywords:
Granular ball computing
KNN
Density peak clustering
Fuzziness
Three-way decision

ABSTRACT

The granular ball k-nearest neighbors (GBKNN) algorithm improves the efficiency and robustness of traditional k-nearest neighbors (KNN) by replacing point input with granular ball. However, in the generation process of granular balls by GBKNN, there may be unbalanced distribution of data points existed in some granular balls, which will cause more classification errors. In addition, the fixed value of k in GBKNN may also reduce the accuracy of classification. In order to address these issues, an adaptive three-way KNN classifier using density-based granular balls is proposed. Firstly, an improved density-based granular ball computing using density peak clustering is presented. This method introduces a refined threshold to subdivide the granular balls. Secondly, a data-driven neighborhood is defined to search the optimal k value and a density-based granular ball KNN (DBGBKNN) algorithm is proposed. Thirdly, by considering the fuzziness of the testing set in the classification process, the density-based granular ball KNN with three-way decision (DBGBKNN-3WD) is constructed. Finally, experimental results verify that DBGBKNN-3WD achieves high comprehensive score and low time complexity while maintaining less fuzziness loss than other algorithms.

1. Introduction

Granular computing (GrC) [1–4] is a method that simulates the human mode of thinking to solve complex tasks. It uses granule formation, transfer, synthesis, and decomposition to process problems at different granulation levels. GrC is an effective method for data mining and fuzzy information analysis. Pedrycz [5] introduces the concept of a map based on information granules and provides a framework for the design of the map. Wang [6] presents a diagram to illustrate the relationship among the three fundamental modes of GrC and analyzes the feasibility for processing big data. Yao [7] explores the relationship between three-way decision (3WD) and GrC. Ouyang and Pedrycz [8] design a characterization of large real data to establish the granular description with two-step.

K-nearest neighbors (KNN) [9] is a supervised learning classifier that uses neighbor relations for classification and prediction. This is a fundamental and straightforward algorithm and it is simple to apply and understand. KNN is generally applied in clas-

* Corresponding author at: College of Computer and Information Science, Southwest University, Chongqing 400715, China.
E-mail addresses: yj530966074@foxmail.com (J. Yang), 34443880@qq.com (J. Kuang), wanggy@cqupt.edu.cn (G. Wang), zhangqh@cqupt.edu.cn (Q. Zhang), yanmin7813@163.com (Y. Liu), liuqun@cqupt.edu.cn (Q. Liu), 1025968052@qq.com (D. Xia), zorro0311@126.com (S. Li), wangxq1230@163.com (X. Wang), wudi.cigit@gmail.com (D. Wu).

https://doi.org/10.1016/j.ins.2024.120858
Received 14 May 2023; Received in revised form 14 May 2024; Accepted 29 May 2024

Available online 3 June 2024

sification [10], regression [11] and outlier detection [12]. Based on the traditional KNN, Zhang [13] presents a multi-label lazy learning approach for classification tasks. Li [14] designs an improved mutual K-nearest-neighbor to optimize the fast density peaks clustering algorithm. To enhance classification performance and diminish sensitivity to the scale of neighborhood space, Gou [15] proposes a new k-nearest centroid neighbors method based on representation coefficient. To evaluate the classification effect, Duan [16] introduces a new classification efficiency index based on augmented non-shared nearest neighbors. However, KNN also has some limitations, such as large amount of computation and memory, sensitivity to imbalance data and the weak interpretation of classification results. This means that KNN relies heavily on the number and dimension of training samples, which is not suitable for dealing with large-scale data sets.

The granular ball k-nearest neighbors (GBKNN) [17] greatly reduces the time complexity of traditional KNN by using granular ball computing (GBC) method. GBC [18–22] is a new method of data abstraction and knowledge extraction based on information balls rather than traditional information granules. Xia [18] proposes a flexible granular ball generation method that accelerates construction of granular balls. Then, a novel theoretical model of rough set [19] is explored for feature reduction on large datasets, called GBNRS. Furthermore, Xia [20] presents a fast adaptive k-means under unbounded conditions by using the granular ball to describe clusters. Additionally, Zhang [23] proposes a granular ball rough set model to classification problems in dynamic mixed-type decision systems. For applying to attribute reduction, Chen [24] proposes a selector using granular balls to reduce the dimensions of data. Peng [25] introduces a robust granular ball model with variable parameters for attribute reduction and classification from a coarse-grained view. Although GBKNN reduces the time complexity of traditional KNN by introducing GBC, there are still some limitations in current GBKNN as follows:

- Although GBC provides an efficient data preprocessing paradigm, it primarily relies on a purity threshold. As a result, some potential errors may be introduced in the subsequent classification process.
- GBKNN is often difficult to select the appropriate k value during classification processes.
- Similar to the traditional KNN, GBKNN still adopts the classic two-way decision approach, which may bring more uncertainty to classification.

It is well known that the density peak clustering (DPC) [26] is able to automatically find the cluster center without relying on any shape, thus achieving the purpose of effective clustering. Moreover, DPC does not require specifying the number of clusters and has higher operational efficiency compared to other clustering algorithms [27][28]. Thus, DPC is an effective tool to further refine the granular balls. To solve the two previous issues mentioned above in GBKNN, a new density-based granular ball computing (DBGBC) method on the basis of DPC is presented in this paper. Furthermore, to search a more suitable k value, this paper proposes a data-driven neighborhood based on optimal deviation, which can automatically select the optimal k value according to data distribution. Then, a density-based granular ball KNN (DBGBKNN) is proposed.

Although DBGBKNN mentioned above can generate finer granular balls and adaptively search a more appropriate k value, it may bring more uncertainty or fuzziness in classification. 3WD [29–33] refers to dividing a universe into three independent regions and each region corresponds to a decision rule. As a generalization of the conventional two-way decision model, 3WD further incorporates a third option and has been universally adopted in different industries and professions. To mine more information and improve the tolerance to errors, Du [34] proposes the three-way clustering algorithm with multi-step strategy depth. Zhan [35] reviews the advances in three-way behavioral decision making (TW-BDM) with hesitant fuzzy information systems. Zhang [36] proposes a novel three-way clustering method (3WC-D), which combines 3WD and feature distribution, achieving clearer clustering results. Zhao [37] develops a multi-criterion 3WD model to process fuzzy probabilistic decision system of picture. Aiming to address conflict analysis in software development, Wang [38] introduces a novel 3WD model that incorporates compound risk preferences and probabilistic linguistic terminologies. Therefore, to reduce the loss of fuzziness caused by using DBGBKNN algorithm in the classification process, the DBGBKNN with 3WD (DBGBKNN-3WD) is proposed.

Based on the aforementioned analyses, the main contributions of this paper are summarized as follows:

- We present an improved GBC based on DPC algorithm that manages the level of granular ball subdivision by incorporating a refined threshold.
- We introduce a data-driven neighborhood based on standard deviation and present a DBGBKNN that incorporates the improved GBC.
- Considering the uncertainty loss caused by the classification process, we construct DBGBKNN-3WD by minimizing the fuzziness loss.

The organization of this paper is outlined: Section 2 introduces basic definitions such as neighborhood rough set, fuzziness, granular ball and 3WD. In section 3, we discuss the disadvantages of GBC and propose a new granular ball generation algorithm based on the DBGBC. Section 4 constructs the DBGBKNN algorithm by combining the data-driven neighborhood with DBGBC and further proposes the DBGBKNN-3WD. The proposed methods are evaluated and compared with other existing methods by experimental simulation analysis in section 5. In Section 6, a detailed summary is given.

2. Preliminaries

In this section, some required concepts related to neighborhood rough set, fuzziness, granular ball and 3WD are presented to simplify the structure of full paper.

Definition 1. (Neighborhood rough set) [39] Given a neighborhood decision system $NS = (U, C \cup D, V, f, \delta)$, $R \subseteq C$ and $X \subseteq U$. Here, X denotes a target concept. The lower and upper approximation sets of X are defined as follows:

$$\underline{NR}(X) = \{ x | \delta_R(x) \subseteq X, x \in U \}, \tag{1}$$

$$\overline{NR}(X) = \{ x | \delta_R(x) \cap X \neq \phi, x \in U \}. \tag{2}$$

Where $\delta_R(x) = \{x_i | d(x, x_i) \leq \delta\}$ and $d(\cdot, \cdot)$ is the Euclidean distance calculation function, δ denotes the neighborhood radius. In NRS, the universe U is divided into positive region, boundary region and negative region. They can be defined as follows:

$$POS_R(X) = \underline{NR}(X), \tag{3}$$

$$BND_R(X) = \overline{NR}(X) - \underline{NR}(X), \tag{4}$$

$$NEG_R(X) = U - \overline{NR}(X). \tag{5}$$

If NS is a neighborhood decision system containing only continuous values, $0 \leq \delta \leq 1$, otherwise $\delta = 0$.

NRS significantly extends the application scope of rough sets, and provides a valuable mathematical method to solve uncertain issues in mixed data. It has generally been applied extensively in attribute reduction [40], feature selection [41] and various types of classification tasks [42].

Definition 2. (The average fuzzy set) [43] Given a neighborhood decision system $NS = (U, C \cup D, V, f, \delta)$, $R \subseteq C$ and $X \subseteq U$. Here, X denotes a target concept. Let $\mu(t)(t \in \delta_R(x))$ represent the menbership of the element t in $\delta_R(x)$ for X, and $\overline{\mu}(x)$ represents the average membership of x, Where

$$\overline{\mu}(x) = \frac{\sum_{t \in \delta_R(x)} \mu(t)}{|\delta_R(x)|}. \tag{6}$$

Then $\forall x_i \in U (i = 1, 2, \ldots, n)$, the average fuzzy set of X is represented as follows:

$$X_R^J = \frac{\overline{\mu}(x_1)}{x_1} + \frac{\overline{\mu}(x_2)}{x_2} + \cdots + \frac{\overline{\mu}(x_n)}{x_n}. \tag{7}$$

Definition 3. (Fuzziness) [44] Given a neighborhood decision system $NS = (U, C \cup D, V, f, \delta)$, M and N denote two fuzzy sets on U, then $F(U)$ denotes a set of all fuzzy subsets on U. If $H : F(U) \to [0, 1]$ satisfies all the following conditions:

(1) $H(M) = 0$, iff $M \in \Phi(U)$, where $\Phi(U)$ is a power set of U;
(2) $H(M) = 1$, iff $\forall x_i \in U$, $M(x_i) = \frac{1}{2}$;
(3) $H(M) \geq H(N)$, iff $\forall x_i \in U$, $N(x_i) \geq M(x_i) \geq \frac{1}{2}$ or $N(x_i) \leq M(x_i) \leq \frac{1}{2}$.

Then $H(\cdot)$ denotes the fuzziness of a fuzzy set.

Definition 4. (Average fuzziness) [44] Given a neighborhood decision system $NS = (U, C \cup D, V, f, \delta)$, $R \subseteq C$ and $X \subseteq U$. Here, X denotes a target concept, and X_R^J denotes the average fuzzy set based on average membership for X. The average fuzziness of x is expressed as $\overline{h}(x) = 4\overline{\mu}(x)(1 - \overline{\mu}(x))$. The average fuzziness of an average fuzzy set X_R^J is constructed as follows:

When U is a discrete universe, the definition of the average fuzziness is represented by:

$$\overline{H}_{X_R^J} = \frac{1}{|U|} \sum_{x \in U} \overline{h}(x). \tag{8}$$

When U is a continuous universe on $[a, b]$, the definition of the average fuzziness is represented by:

$$\overline{H}_{X_R^J} = \frac{1}{|b - a|} \int_a^b \overline{h}(x). \tag{9}$$

Definition 5. (Three-way decision rules) [29,30] Given a threshold pair $[\alpha, \beta]$. Let x denote an element and $\overline{\mu}(x)$ denotes the average membership of x. The three-way decision rules can be expressed as follows:

(1) $x \in POS(\alpha,\beta)$, iff $\overline{\mu}(x) \geq \alpha$;
(2) $x \in BND(\alpha,\beta)$, iff $\beta < \overline{\mu}(x) < \alpha$;
(3) $x \in NEG(\alpha,\beta)$, iff $\overline{\mu}(x) \leq \beta$.

When determining a threshold pair $[\alpha,\beta]$, various elements can be judged into corresponding regions according to their average membership.

Definition 6. (The center and radius of granular ball) [17] Given a real dataset D and a granular ball B on D. C_B denotes the center of B. Then, r_B denotes the radius of B. Let N be the number of sample points in B and $E_i (i = 1, \ldots, N)$ be a point in B, we have:

$$C_B = \frac{1}{N}\sum_{i=1}^{N} E_i, \tag{10}$$

$$r_B = \frac{1}{N}\sum_{i=1}^{N} |E_i - C_B|. \tag{11}$$

The purpose of defining the radius r_B as the average distance is to avoid the possible influence of outlier samples in the subsequent processing.

Definition 7. (The distance from a point to a granular ball) [17] Given a real dataset D and a granular ball B on D. C_B denotes the center of B. Then, r_B denotes the radius of B. Let O be a test point, the distance from O to B is denoted as $dis(O,B)$ as follows:

$$dis(O,B) = \begin{cases} dis(O,C_B) - r_B & \text{, if } dis(O,C_B) - r_B > 0 \\ 0 & \text{, else} \end{cases} \tag{12}$$

Definition 8. (The label of granular ball) [17] Given a real dataset D and a granular ball B on D. The label of B matches the label of most of elements in B.

3. The density-based granular ball generation

In this section, we first analyze the differences between the granular ball and traditional information granules, i.e., equivalence class, neighborhood granule and fuzzy rough granule. Moreover, we further present the advantages of granular ball compared with these three granules in classification tasks. Then, the algorithm for generating granular balls based on GBC is introduced, and its advantages and disadvantages are discussed.

Traditional information granules employ discrete regions or subsets for data granulation. Pawlawk rough sets use equivalence classes to represent knowledge, but it fails to handle continuous data; neighborhood rough sets and fuzzy rough sets can handle continuous data, but they are computational time and fail to represent knowledge. In contrast, GBC exhibits the characteristics of efficiency, robustness, and interpretability by replacing point input with granular balls. In terms of efficiency, granular balls improve the processing efficiency of large-scale datasets through multi-granularity computation. Additionally, it demonstrates robustness by effectively handling various types and shapes of data, adapting to non-convex data, noisy data, and outliers. Furthermore, granular balls offer interpretability by utilizing spherical granules and visualizing the spherical granules to intuitively showcase the similarities and associations among data objects. Therefore, the granular ball outperforms other information granules in terms of comprehensive data representation, improved classification accuracy, intuitive and flexible parameter settings, and efficient computational performance.

As shown in Algorithm 1, the generation method of granular balls proposed by Xia [17] provides a tool for preprocessing datasets. From the literature [17], in the worst case, the time complexity of Algorithm 1 approaches $O(kn^2)$, where k denotes iteration number, and n denotes sum of elements in D. The purity threshold is an important parameter to control the degree of granularity in the division of granular balls. By calculating the proportion of the majority label, the purity of a granular ball can be determined. Once the purity of granular balls is unable to reach the purity threshold, the granular balls will be divided again. This means that the higher the purity threshold, the higher the purity of the final generated granular balls. On one hand, when the purity threshold is less than 1 may result in elements belong to different classes being classified under the same label. On the other hand, when the purity threshold is equal to 1 may lead to an unbalanced distribution of data points within the partitioned granular ball. This means that Algorithm 1 need to be further improved.

In 2014, Rodriguez et al. [26] proposed the DPC by searching the density peak points (DPPs) as the cluster centers. This algorithm automatically discovers cluster centers and further efficiently classifies data of arbitrary shape. DPC introduces two important parameters to identify DPPs: LD for local density and RD for relative distance. A point is considered as a density peak point only if both of its parameters are large enough after calculation. However, traditional DPC does not give a threshold to judge whether two parameters are large enough. Therefore, this paper introduces a new parameter to control the degree of refinement, called the refined threshold. By using the refined threshold RT, the local density threshold LDT and relative distance threshold RDT are calculated as follows:

Algorithm 1: GBC [17].

Input: A real dataset D, a purity threshold p
Output: The set of granular balls GB_list

$GB\_list = \{\}$;
Function `Split_GB(B)`:
  if *the purity of* $B >= p$ **then**
    Add B to the GB_list;
  end
  else
    B_1, $B_2 \leftarrow B$ is subdivided into two granular balls by 2-means clustering algorithm;
    Split_GB(B_1);
    Split_GB(B_2);
  end
Function `Main(D)`:
  Trun the dataset D to a granular ball B;
  Split_GB(B);
  return GB_list

$$LDT = min(LD) + (max(LD) - min(LD)) * RT, \tag{13}$$

$$RDT = min(RD) + (max(RD) - min(RD)) * RT. \tag{14}$$

Where LD denotes the local density, RD denotes the relative distance, and $0 < RT < 1$. Obviously, larger values of RT result in larger values of LDT and RDT, which means fewer DPPs. Therefore, the DPC with refined threshold can effectively determine whether the distribution of data points is balanced.

In this paper, a new density-based granular ball computing method called DBGBC by using the DPC with refined threshold is proposed. Firstly, to avoid that elements belonging to different classes are classified under the same label, the purity threshold is fixed to 1 for generating granular balls. Secondly, the DPC with refined threshold is used to determine whether the distribution of data points is balanced. When the number of DPPs calculated by DPC in a granular ball is less than half of sum of elements in the granular ball and greater than 2, the distribution of elements is unbalanced. Therefore, this granular ball will be further subdivided by 2-means algorithm.

With this method, the generated granular balls are further refined by setting a refined threshold. The corresponding algorithm of DBGBC is shown in Algorithm 2. In this algorithm, we introduce the refined threshold to assess the balanced distribution of elements within each granular ball, further enhancing the quality of granular balls. The refined threshold is used to measure the balance of element distribution within a granular ball. If the distribution within a granular ball is imbalanced, we further split it into new granular balls, ensuring a better representation of data features and relationships. The introduction of the refined threshold aims to enhance the representability and precision of the classifiers based on granular ball computing.

Algorithm 2: DBGBC.

Input: GB_list obtained by Algorithm 1, the refined threshold RT
Output: The list of density-based granular balls $DBGB\_list$

$DBGB\_list = \{\}$;
Function `Split_DBGB(B)`:
  Calculate the number of DPPs of B calculated by DPC using the refined threshold RT;
  if *the number of DPPs of* B *is less than 2 or larger than half of elements in* B **then**
    Add B to the $DBGB\_list$;
  end
  else
    B_1, $B_2 \leftarrow B$ is subdivided into two granular balls by 2-means clustering algorithm;
    Split_DBGB(B_1);
    Split_DBGB(B_2);
  end
Function `Main(GB_list)`:
  for GB_i *in* GB_list **do**
    Split_DBGB(GB_i)
  end
  return $DBGB\_list$

The efficiency of Algorithm 2 is explained:

- Getting B obtained by Algorithm 1 requires $O(kn^2)$ time.
- Calculating the number of DPPs and deciding whether to split the granular ball further using the 2-means requires at most k iterations requires $O(kn)$ time.

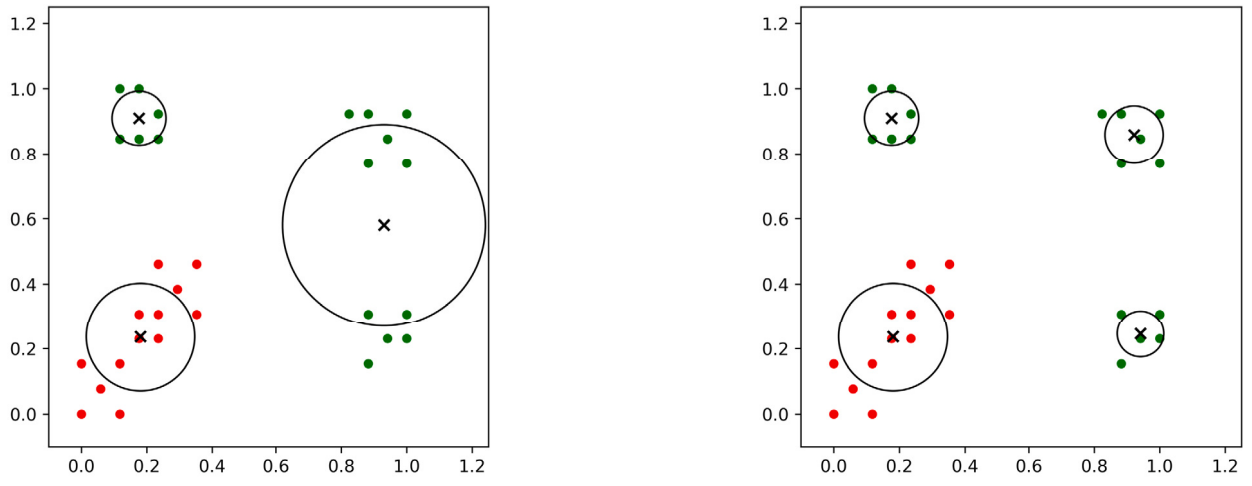

Fig. 1. Results generated granular balls using GBC and DBGBC on the same dataset.

Where k represents the iteration number, and sum of elements in D is represented by n. Thus, the time complexity of the Algorithm 2 is still $O(kn^2)$.

DBGBC realizes the further division of the original granular balls by using the refined threshold. By setting the purity threshold to 1 and using the refined threshold, a more balanced distribution of elements in each granular ball is achieved. This means that DBGBC is able to divide more reasonable granular balls for subsequent processing. Thus, DBGBC is more suitable for classification tasks than GBC.

Fig. 1 shows the results generated granular balls using GBC and DBGBC on the same dataset. Fig. 1a shows that GBC is used to divide the elements in original dataset into three granular balls, but it is obviously observed that unbalanced distribution of elements within the rightmost granular ball. Fig. 1b shows that the results by using DBGBC. Obviously, two reasonable granular balls are obtained by partitioning the original rightmost granular ball again.

4. Adaptive three-way KNN classifier using density-based granular ball

GBKNN [17] is proposed to enhance the performance of KNN, as shown in Algorithm 3. Because of its efficiency, GBKNN is suitable for large-scale data classification tasks without requiring the selection of parameter k. If the sum of elements in GB_list is represented by n, the time complexity of GBKNN is denoted by $O(n)$. However, GBKNN adopts the GBC theory and therefore inherits its shortcomings, resulting in a loss of accuracy. The DBGBC improved the GBC theory by proposing a density-based granular balls generation algorithm. Additionally, to improve efficiency, GBKNN always selects the label of the nearest granular ball ($k = 1$) during classification, which also leads to a decline in accuracy. To address this issue, an adaptive computing paradigm for searching the optimal value of k is introduced.

Algorithm 3: GBKNN [17].

Input: The list of granular balls GB_list obtained by Algorithm 1, a queried point O
Output: the label of $GB_{nearest}$
Function `Main(`GB_list,O`)`**:**
  for GB_i *in* GB_list **do**
    Calculate the distance from O to GB_i;
  end
  $GB_{nearest}$ ← Search the nearest granular ball to O;
  return the label of $GB_{nearest}$

4.1. The data-driven neighborhood

This section provides the definition of optimal deviation, shown in Definition 9, to search for the optimal value of k.

Definition 9. (Optimal deviation) [45] Let $A_{n-1} = \{a_1, a_2, \ldots, a_{n-1}\}$ and $A_n = \{a_1, a_2, \ldots, a_{n-1}, a_n\}$ be two sets of real numbers. $\overline{A}_{n-1} = \frac{1}{n-1}\sum_{i=1}^{n-1} a_i$ and $\overline{A}_n = \frac{1}{n}\sum_{i=1}^{n} a_i$ denote the average values of $\overline{A}_{n-1}$ and $\overline{A}_n$, respectively. $S_{n-1} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n-1}(a_i - \overline{A}_{n-1})^2}$ and

$S_n = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(a_i - \overline{A}_n)^2}$ are the standard deviations of A_{n-1} and A_n, respectively. The optimal deviation Z_n of element a_n in A_n is defined as $Z_n = S_n - S_{n-1} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(a_i - \overline{A}_n)^2} - \sqrt{\frac{1}{n-1}\sum_{i=1}^{n-1}(a_i - \overline{A}_{n-1})^2}$.

According to Definition 9, A_n contains one additional element a_n compared to A_{n-1}. The addition of element a_n to A_{n-1} may change its dispersion degree. The value of optimal deviation reflects the impact of the new added value a_n on A_{n-1}. The element a_n is considered to have a negative effect when $Z_n > 0$, a positive effect when $Z_n < 0$, and no effect when $Z_n = 0$.

The optimal deviation metric is designed to evaluate the dispersion or spread of a set of points by quantifying the impact of adding a new point. Specifically, it measures how much the standard deviation of the set changes when a new point is added. If adding a new point increases the standard deviation, this indicates the point is further away from the existing set and causes the points to become more dispersed. In this case, the optimal deviation is positive. On the other hand, if adding the new point decreases the standard deviation, this suggests the point reduces the dispersion by being closer to the existing points. Here the optimal deviation is negative. The key idea is that for a set of points closest to a query point, adding a more distant point will increase dispersion and give a positive optimal deviation. This indicates that the optimal neighborhood has been exceeded. Therefore, by incrementally adding more distant points and monitoring the optimal deviation, the data-driven approach can dynamically determine when the neighborhood starts to become too dispersed around the query point. This gives the optimal local k value for that specific neighborhood. The optimal deviation metric provides a quantitative dispersion measure to evaluate new points against the existing neighborhood. This is the foundation that enables the data-driven technique to adaptively determine neighborhood size and optimize k for each local region of the feature space.

Based on Definition 9, the data-driven neighborhood is proposed for selecting the optimal value of k in Definition 10.

Definition 10. (The data-driven neighborhood) Given a point O and a set of points S that does not contain O. Based on the distance between point O and each point in S, the points in S are sorted in ascending order to generate a new set $D = \{x_1, x_2, x_3, \ldots, x_n\}$. The set to be tested is initialized as $TS = \{O\}$, and $x_i (i = 1, 2, \ldots, n)$ is added to TS successively to calculate the optimization deviation until the optimal deviation is larger than 0. Finally, the TS generated is called the data-driven neighborhood of the point O.

An example is provided below to further explain the data-driven neighborhood.

Example 1. Given a point O and a set of points S that does not contain O. Suppose that the distance from point O to each point in S is arranged in ascending order to generate the set $DIS = \{1, 1, 1.2, 5, 12\}$, and the points in S are sorted in ascending order to generate a new set $D = \{x_1, x_2, x_3, x_4, x_5\}$. The set to be tested is initialized as $TS = \{O\}$, and x_i is added to TS successively to calculate the optimal deviation as follows:

- Add x_1 to TS and calculate the standard deviation of TS. $S_1 = \sqrt{\frac{1}{2}((0 - \frac{0+\mathbf{1}}{2})^2 + (1 - \frac{0+\mathbf{1}}{2})^2)} = 0.5$. Then put the result into the standard deviation set $STD = \{0.5\}$.
- Add x_2 to TS and calculate the standard deviation of TS. $S_2 = \sqrt{\frac{1}{3}((0 - \frac{0+1+\mathbf{1}}{3})^2 + 2 * (1 - \frac{0+1+\mathbf{1}}{3})^2)} = 0.4714$. Due to $Z_2 = S_2 - S_1 < 0$, x_2 is considered to have a positive effect. Then put the result into the standard deviation $STD = \{0.5, 0.4714\}$.
- Add x_3 to TS and calculate the standard deviation of TS. $S_3 = 0.4690$. Due to $Z_3 = S_3 - S_2 < 0$, x_3 is also considered to have a positive effect. Then put the result into the standard deviation $STD = \{0.5, 0.4714, 0.4690\}$.
- Add x_4 to TS and calculate the standard deviation of TS. $S_4 = 1.7315$. Due to $Z_4 = S_4 - S_3 > 0$, x_4 is considered to have a negative effect. Then removes point x_4 from TS.

Now, the set $TS = \{O, x_1, x_2, x_3\}$ is the data-driven neighborhood of O about the set S.

In Example 1, the three nearest points are selected to form the data-driven neighborhood of a point O, so the optimal value of k is equal to 3. Thus, Definition 10 provides an effective solution to search the optimal value of k.

4.2. Fast granular ball KNN classifier

Combined with DBGBC and the data-driven neighborhood mentioned above, we further propose the DBGBKNN algorithm, as shown in Algorithm 4.

If the sum of elements in $DBGB\_list$ is represented by n, the time complexity of DBGBKNN is denoted by $O(n)$, which is explained:

- To calculate the distance from O to $DBGB_i$, all granular balls must be traversed, resulting in a time complexity of $O(n)$.
- The time complexity of calculating the data-driven neighborhood of O is $O(nlogn)$.
- Counting the labels of all elements in the data-driven neighborhood of O takes $O(n)$ time.

Algorithm 4: DBGBKNN.

Input: The list of density-based granular balls $DBGB\_list$ obtained by Algorithm 2, a queried point O
Output: The label of O
Function `Main(`$DBGB\_list$,O`)`:
  for $DBGB_i$ *in* $DBGB\_list$ **do**
    Calculate the distance from O to $DBGB_i$;
  end
  Calculate the data-driven neighborhood of O;
  Count the labels of all elements in the data-driven neighborhood;
  return the label with the largest number

4.3. Constructing DBGBKNN with three-way decision based on fuzziness

When DBGBKNN is used for binary classification, the label of a test point is derived from granular balls in its corresponding data-driven neighborhood. However, due to the potential uncertainty of data-driven neighborhood of a test point, employing a classic two-way decision approach may result in errors or inefficiencies. According to further analysis and judgment of undetermined parts, the 3WD approach [29] has the potential to reduce errors and strengthen the reasonableness of decisions. Current research on 3WD focuses primarily on calculating thresholds based on predefined risk parameters to achieve with minimum cost. However, in practical applications, risk parameters are often subjectively determined based on expert experience. This makes it challenging to accurately obtain the necessary risk parameters in 3WD. To solve this problem, uncertainty measure is introduced into 3WD, which provides a new perspective for 3WD theory. Firstly, based on fuzziness invariance, we construct a data-driven 3WD model of neighborhood rough set acting on the training set.

Definition 11. Given a neighborhood decision system $NS = (U, C \cup D, V, f, \delta)$, where $R \subseteq C$ and $X \subseteq U$. Let $\overline{\mu}(x)$ represent the average membership of $x \in \delta_R(x)$ calculated from $\delta_R(x)$. X denotes a target concept, and X_R^J denotes the average fuzzy set based on average membership for X. A mapping $\psi : X_R^J \to \{0, [\beta, \alpha], 1\}$ can be used to represent the frame of the three-way approximations of X_R^J with a shadowed set. Here, the mapping ψ is from X_R^J to the set $\{0, [\beta, \alpha], 1\}$ and $\psi(X_R^J)$ is defined by:

$$\psi(X_R^J) = \begin{cases} 0, & \overline{\mu}(x) \le \beta \\ [\beta, \alpha], & \beta < \overline{\mu}(x) < \alpha \\ 1, & \overline{\mu}(x) \ge \alpha \end{cases} \tag{15}$$

According to X, the average membership of $x \in U$ is calculated, and the average fuzzy set X_R^J is formed. Then, X_R^J is further three-way approximated as follows:

(1) When $\overline{\mu}(x) \le \beta$, $\overline{\mu}(x)$ is reduced to 0, indicating that assigning x to the negative region results in the minimum uncertainty loss.
(2) Conversely, when $\overline{\mu}(x) \ge \alpha$, $\overline{\mu}(x)$ is elevated to 1, indicating that assigning x to the positive region results in the minimum uncertainty loss.
(3) When $\beta < \overline{\mu}(x) < \alpha$, $\overline{\mu}(x)$ is transformed into $[\beta, \alpha]$, which means that assigning x to the boundary region will result in the minimum uncertainty loss.

Herein, when $\beta < \overline{\mu}(x) < \alpha$, the membership degrees are transformed from $\overline{\mu}(x)$ to $[\beta, \alpha]$, and corresponding elements are characterized by a great uncertainty. The following equation can be used to calculate fuzziness-based uncertainty of $\psi(X_R^J)$:

$$\overline{H}_{\psi(X_R^J)} = \frac{4\left|\, x \in U \,|\, \beta < \overline{\mu}(x) < \alpha \right|}{|U|} \int_a^b \overline{\mu}(x)(1 - \overline{\mu}(x)) d(\overline{\mu}(x)). \tag{16}$$

Secondly, to get the minimum uncertainty loss, $[\beta, \alpha]$ for the three-way classification is calculated from the perspective of fuzziness invariance.

Definition 12. Given a neighborhood decision system $NS = (U, C \cup D, V, f, \delta)$, where $R \subseteq C$ and $X \subseteq U$. Let $\overline{\mu}(x)$ represent the average membership of $x \in \delta_R(x)$ calculated from $\delta_R(x)$. X denotes a target concept, and X_R^J denotes the average fuzzy set based on average membership for X. The improved lower and upper approximation sets of X are defined by:

$$\underline{NR}(X) = \{\, x \in U \,|\, \overline{\mu}(x) \ge \alpha \,\}, \tag{17}$$

$$\overline{NR}(X) = \{\, x \in U \,|\, \overline{\mu}(x) > \beta \,\}. \tag{18}$$

Accordingly, the rules for dividing the three decision regions according to $\overline{\mu}(x)$ are defined by:

$$POS_R^{(\beta,\alpha)}(X) = \underline{NR}(X) = \{x \in U \,|\, \overline{\mu}(x) \ge \alpha\}, \tag{19}$$

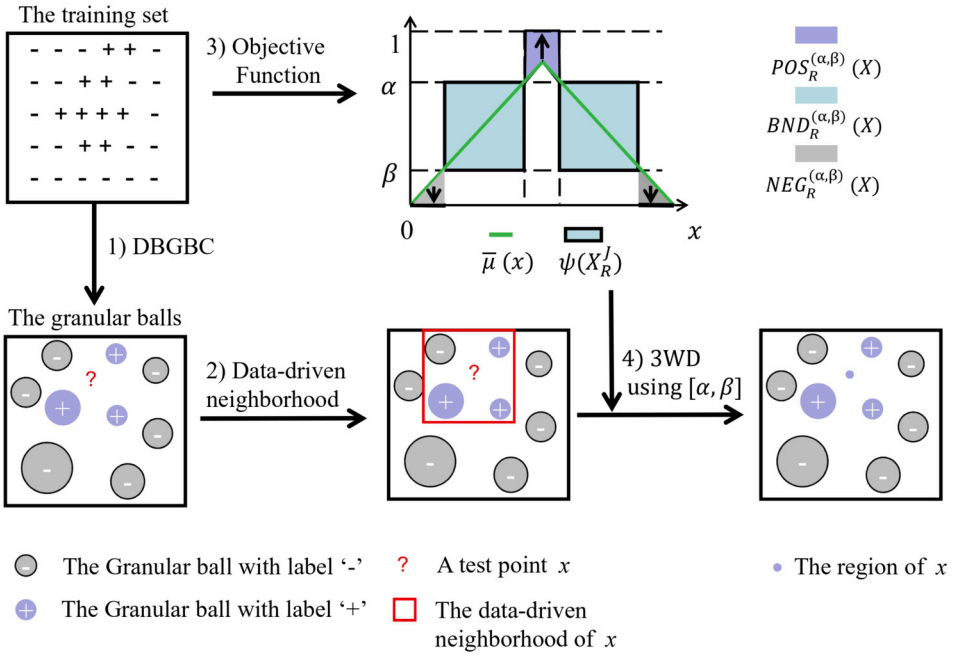

Fig. 2. The interpretation of the construction process of DBGBKNN-3WD from the perspective of fuzziness.

$$NEG_R^{(\beta,\alpha)}(X) = U - \overline{NR}(X) = \{x \in U | \overline{\mu}(x) \le \beta\}, \tag{20}$$

$$\begin{aligned} BND_R^{(\beta,\alpha)}(X) &= \overline{NR}(X) - \underline{NR}(X) \\ &= \{x \in U | \beta < \overline{\mu}(x) < \alpha\}. \end{aligned} \tag{21}$$

The three-way approximation of X_R^J using different $[\beta,\alpha]$ leads to different degrees of fuzziness loss. Therefore, in order to calculate the optimal thresholds that minimizes the fuzziness loss, the objective function is defined by:

$$\begin{aligned} &\underset{0\le\beta\le\alpha\le1}{\operatorname{argmin}} \left| \overline{H}_{\psi(X_R^J)} - \overline{H}_{X_R^J} \right|, \\ &s.t.\ 0 \le \beta \le 0.5, 0.5 \le \alpha \le 1. \end{aligned} \tag{22}$$

Where

$$\begin{aligned} \overline{H}_{\psi(X_R^J)} - \overline{H}_{X_R^J} = \frac{4}{|U|}(&\left|\{x \in U | \beta < \overline{\mu}(x) < \alpha\}\right| * \\ &\int_\beta^\alpha \overline{\mu}(x)(1-\overline{\mu}(x))d(\overline{\mu}(x)) - \sum_{x\in U} \overline{\mu}(x)(1-\overline{\mu}(x))). \end{aligned} \tag{23}$$

According to the objective function of Definition 12, based on Algorithm 4, two decision thresholds are introduced to calculate the minimum fuzziness loss on the training set, and then apply it to the three-way classification of the testing set. The details are shown in Algorithm 5.

In addition, Fig. 2 shows the decision process of DBGBKNN-3WD. The step 1 is to apply DBGBC on the training set to generate a set of granular balls, which consists of different sized granular balls containing the label information. For a test point x in the testing set, its corresponding data-driven neighborhood is identified through the step 2, which is the most suitable k value for x. Based on the data-driven neighborhood, the membership degree of x is then calculated. The step 3 is to determine the 3WD threshold of the original training set by minimizing the fuzziness loss. In the final step, the membership of x is compared with the calculated three-way decision threshold to classify x. As illustrated in Fig. 2, the 3WD threshold is assumed as $[0.2, 0.7]$. Since the membership degree of x calculated using its data-driven neighborhood is 0.75 which is greater than 0.7, x belongs to positive region.

If the sum of elements in $DBGB\_list$ is represented by n, the time complexity of DBGBKNN-3WD is denoted by $O(n)$, which is explained as follows:

- To calculate the distance from O to $DBGB_i$, all granular balls must be traversed, resulting in a time complexity of $O(n)$.
- The time complexity of calculating the data-driven neighborhood of O is $O(nlogn)$.
- To calculate the membership of point O in the data-driven neighborhood takes $O(n)$ time.

Algorithm 5: DBGBKNN-3WD based on fuzziness.

Input: The list of density-based granular balls $DBGB\_list$ obtained by Algorithm 2, a queried point O, the decision thresholds α and β
Output: The region of O

Function `Main(`$DBGB\_list$,O`)`**:**
  $region = \{\}$;
  for $DBGB_i$ *in* $DBGB\_list$ **do**
    Calculate the distance from O to $DBGB_i$;
  end
  Calculate the data-driven neighborhood of point O;
  Calculate the membership of point O ;
  if *the membership of point* $O \geq \alpha$ **then**
    $region \leftarrow pos$;
  end
  else if *the membership of point* $O \leq \beta$ **then**
    $region \leftarrow neg$;
  end
  else
    $region \leftarrow bnd$;
  end
  return $region$

Table 1
The description table of datasets.

	Datasets	Characteristics	Instances	Attributes
D1	Hcv	Categorical	596	10
D2	Breast	Real	699	9
D3	Naive-Bayes	Real	955	2
D4	Endgame	Categorical	958	9
D5	Banknote	Real	1371	4
D6	Abalone	Categorical	4177	8
D7	Banana	Categorical	5330	2
D8	Mushroom	Real	8124	22
D9	Dry-Bean	Real	13611	16

5. Experiments

In this section, four research points (RPs) with respect to the proposed methods and algorithms are verified one by one as follows:

- RP. 1. The proposed DBGBC method effectively subdivides the granular balls generated based on GBC, as demonstrated by quantitative results obtained from experiments.
- RP. 2. The data-driven neighborhood method significantly improves the accuracy of GBKNN, as supported by experimental results with specific metrics.
- RP. 3. The 3WD model reduces the fuzziness loss of DBGBKNN compared to the traditional two-way decision, as demonstrated by quantitative results obtained from experiments.
- RP. 4. The DBGBKNN-3WD algorithm not only improves the accuracy of GBKNN, but also outperforms other KNN algorithms based on neighborhood in terms of comprehensive score, as measured by specific evaluation criteria in experiments.

5.1. General settings

Datasets. Nine UCI public datasets are used for benchmark testing in the experiment, as shown in Table 1. For each verification experiment, a ten-fold cross-validation method is adopted.

Evaluation Metrics. For four different experiments, different evaluation methods are adopted.

For the first experiment, the effectiveness of DBGBC needs to be verified. Thus, two evaluation metrics are set: one is the number of granular balls, and another is the generation time. The number of granular balls reflects level of dataset subdivision, and the generation time reflects the efficiency of the two different methods.

For the second experiment, the validity of the data-driven neighborhood needs to be confirmed. The original GBKNN and the GBKNN using data-driven neighborhood are used for classification on nine datasets. Therefore, two metrics are set for evaluation: classification accuracy and classification time. Set U_{test} to denote the number of elements that have been classified in the testing set, and R_{test} to denote the number of correctly classified elements. The accuracy is calculated using the following formula:

$$accuracy = \frac{R_{test}}{U_{test}}. \tag{24}$$

For the third experiment, the validity of 3WD needs to be verified. Since this 3WD is constructed from the perspective of minimum fuzziness loss, thus the fuzziness loss is considered as the evaluation metric. The specific calculation formula for fuzziness loss is shown in Formula (23).

For the last experiment, the validity of DBGBKNN-3WD needs to be verified. The same evaluation metrics as in the second experiment are used. Additionally, three common machine learning metrics, namely F1 score ($F1$), precision (P), and recall (Re), are also calculated for further evaluation. In binary classification problems, $F1$ refers to the harmonic average between P and Re. The larger the value of the three metrics, the better the classification effect. P indicates the proportion of elements predicted to be positive that are actually correct, while Re indicates the proportion of elements that are actually positive and correctly identified. Suppose that sum of positive elements predicted correctly is tp, sum of negative elements predicted correctly is tn, sum of positive elements predicted incorrectly is fp, and sum of negative elements predicted incorrectly is fn, then the calculation formulas of metrics P, Re, and $F1$ are as follows:

$$P = \frac{tp}{tp + fp}, \tag{25}$$

$$Re = \frac{tp}{tp + fn}, \tag{26}$$

$$F1 = \frac{2 * P * Re}{P + Re}. \tag{27}$$

Implementation Detail. In RP. 1, GBC is compared with the proposed DBGBC method.

In RP. 2, GBKNN is compared with the GBKNN algorithm using data-driven neighborhood which is simplified to GBKNN-II.

In RP. 3, DBGBKNN and DBGBKNN-3WD are compared to verify the fuzziness loss of 3WD based on fuzziness.

In RP. 4, ablation experiments for the three submodules DBGBC, data-driven neighborhood and 3WD are conducted. Wherein, GBKNN-I represents replacing the original GBC in GBKNN with DBGBC, and GBKNN-II represents changing the selection of k value from fixed $k = 1$ to $k = k^*$ determined by using data-driven neighborhood on the basis of GBKNN. GBKNN-3WD represents the addition of data-driven neighborhood and 3WD on the basis of GBKNN. Then, DBGBKNN adopts DBGBC and data-driven neighborhood. Finally, DBGBKNN-3WD represents an algorithm after adding 3WD based on DBGBKNN.

In addition to these KNNs mentioned earlier, we also introduced four other KNN algorithms: the multi-label learning algorithm (ML-KNN) [13], the nearest centroid algorithm (NC) [46], fuzzy-rough KNN with tolerance (FRNN-T) [47], and fuzzy-rough KNN with gaussian kernel function (FRNN-G) [47]. Here are the detailed descriptions of these four algorithms:

- ML-KNN: ML-KNN employs Bayesian conditional probability to determine the existence or absence of a label for each test element on the basis of its k nearest neighbors within training set. The core idea of ML-KNN is similar to KNN that the classification of a test element is based on finding k nearest neighbors. However, ML-KNN incorporates probabilistic reasoning to estimate the likelihood of label existence, making it a more sophisticated approach compared to traditional KNN.
- NC: NC is a centroid-based classification algorithm. It divides the elements into several classes based on their similarity, with higher similarity observed within the same class and lower similarity observed between different classes. NC leverages the concept of centroids to determine the class of a test element on the basis of its similarity to centroids of different classes.
- FRNN-T: FRNN-T is an extension of KNN that incorporates fuzzy-rough set theory with tolerance. It utilizes the concept of tolerance to handle uncertainty in the classification process. The fuzzy relationship between a test element and its k nearest neighbors is determined based on the tolerance value, allowing for a more flexible classification decision. By incorporating the fuzzy-rough set theory, FRNN-T can handle imprecise and uncertain data more effectively.
- FRNN-G: FRNN-G is another variant of fuzzy-rough KNN that utilizes a gaussian kernel function to determine the fuzzy relationship between a test element and its k nearest neighbors. The gaussian kernel function measures the similarity between elements based on their distance, assigning higher weights to closer neighbors and lower weights to farther neighbors. This approach allows for a smooth transition of influence from neighboring elements, providing a more continuous and robust classification decision.

In Table 3, we summarize all the algorithms for experimental comparison and give a brief description of each algorithm.

Hardware Configuration. All of experiments is operated on a system with an Intel i5-10500 CPU (6 cores, clock speed of 3.1 GHz), 16 GB of DDR4 running memory (dual-channel configuration), and Windows 10 64-bit OS.

5.2. GBC VS DBGBC (RP. 1)

In the first experiment, Fig. 3 illustrates different number of granular balls separately generated by GBC and DBGBC using different refined thresholds for datasets. It also shows the time consumed during the granular ball generation process when the purity value is set to 1.

As shown in Fig. 3, for a given dataset and different refined thresholds, DBGBC consistently generates a larger number of granular balls compared to GBC. This trend is observed across datasets of different sizes. In most cases, the generation time of DBGBC is similar to that of GBC, and the generation time of GBC is of the same order of magnitude as DBGBC. This suggests that DBGBC has the ability to further refine granular balls. Therefore, the DBGBC method effectively subdivides the granular balls generated based on GBC, as demonstrated by quantitative results obtained from experiments.

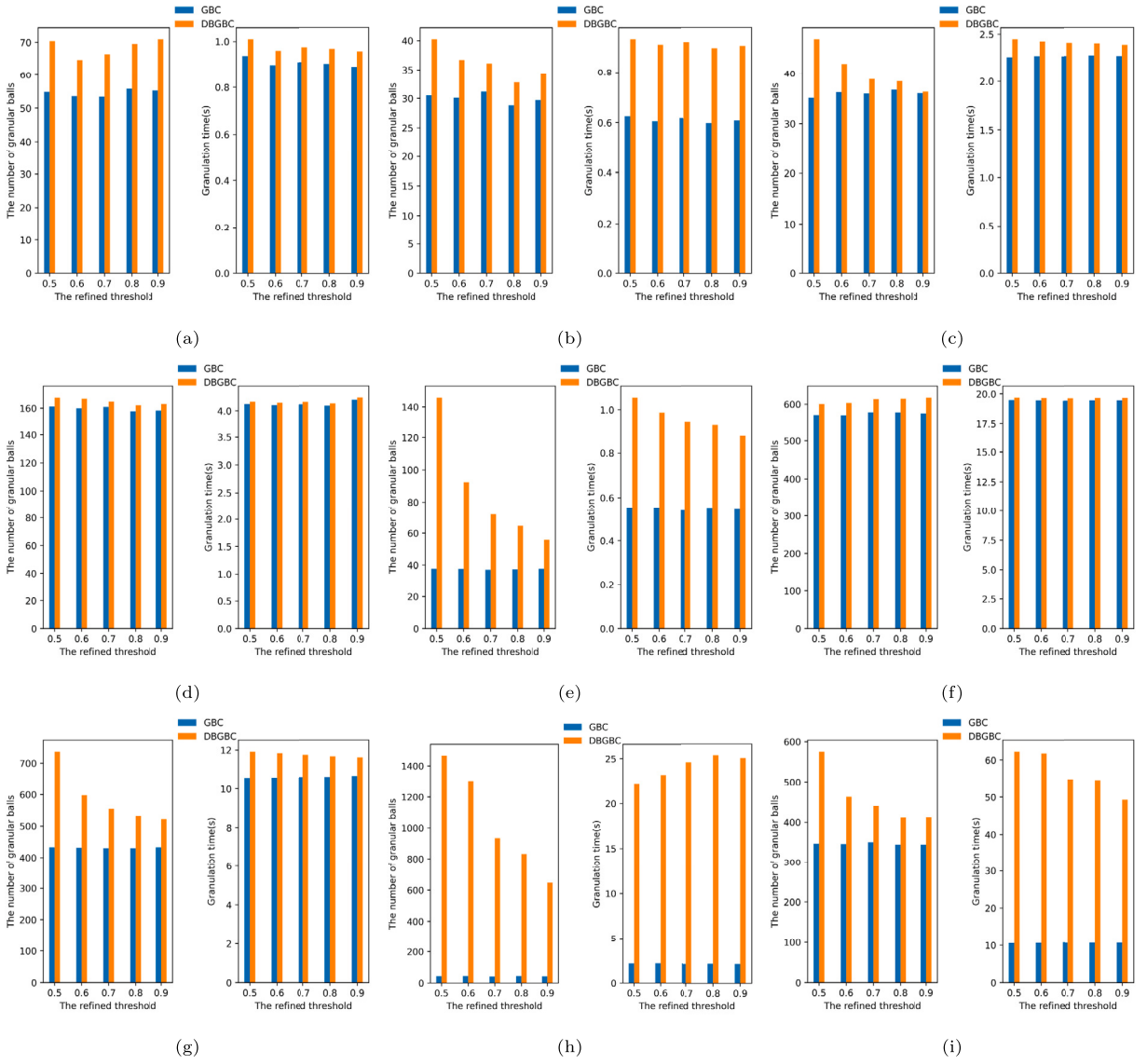

Fig. 3. The number of granular balls and the granulation time generated by GBC and DBGBC when the refined threshold is 0.5 to 0.9.

5.3. GBKNN VS GBKNN-II (RP. 2)

When using the GBKNN algorithm for classification, Only the nearest granular ball is considered for judging a test point, typically with $k = 1$. The data-driven neighborhood approach aims to search the optimal value for k based on the data distribution. Let k^* denote the value of k determined by the data-driven neighborhood, i.e., $k = k^*$. Table 2 presents the classification accuracy on different datasets using two different schemes for searching the value of k.

As shown in Table 2, the accuracy of GBKNN is generally improved on some datasets when the data-driven neighborhood approach is used to search the value of k. The reason is that when the data-driven neighborhood is used alone, uncertainty loss may be caused, which will lead to the reduction of classification accuracy. Therefore, although GBKNN-II only performs well when classifying some datasets, it provides ideas for further construction of the 3WD model.

Furthermore, we also analyze the potential time costs associated with using data-driven neighborhoods for selecting an appropriate k value. Table 4 illustrates the time consumed by using data-driven neighborhood approach on nine different datasets.

As shown in Table 4, we observe that when classifying with $k = k^*$, there is an additional time cost compared to using $k = 1$. However, this time cost changing from $O(n)$ to $O(n \log n)$ is generally acceptable for most classification tasks. Thus, we demonstrate that utilizing data-driven neighborhood to select the value of k can indeed improve the classification accuracy in some cases, despite the introduction of some time cost. Therefore, the data-driven neighborhood method improves the accuracy of GBKNN under certain conditions, as supported by experimental results with specific metrics.

Table 2
Classification accuracy of GBKNN and GBKNN-II.

Datasets	GBKNN($k = 1$)	GBKNN-II($k = k^*$)
D1	0.867033898	**0.877231638**
D2	0.961387163	**0.969979296**
D3	0.920484848	**0.932535353**
D4	0.707861842	**0.707861842**
D5	0.999270073	0.978842695
D6	0.650217434	**0.653800213**
D7	0.889056604	0.885283019
D8	0.98473785	0.861027545
D9	0.963857231	0.962976552

Table 3
Summary of comparision models.

RP	Method or Algorithm	Description
1	GBC	2-means
	DBGBC	2-means and DPC
2	GBKNN	GBC and $k = 1$
	GBKNN-II	GBC and $k = k^*$
3	DBGBKNN	DBGBC and $k = k^*$
	DBGBKNN-3WD	DBGBKNN and 3WD
4	KNN	Traditional KNN
	ML-KNN [13]	Conditional probability
	NC [46]	Centroid-based
	FRNN-T [47]	Tolerance-based
	FRNN-G [47]	Gaussian kernel-based
	GBKNN [17]	GBC and $k = 1$
	GBKNN-I	DBGBC and $k = 1$
	GBKNN-II	GBC and $k = k^*$
	GBKNN-3WD	GBKNN-II and 3WD
	DBGBKNN	DBGBC and $k = k^*$
	DBGBKNN-3WD	DBGBKNN and 3WD

Table 4
The required time of GBKNN and GBKNN-II.

Datasets	GBKNN($k = 1$)	GBKNN-II($k = k^*$)
D1	0.04426174	0.04507522
D2	0.02650604	0.02788872
D3	0.04839858	0.05017558
D4	0.21575751	0.21246095
D5	0.0676181	0.06958162
D6	3.29948898	3.33018495
D7	3.28217726	3.31035479
D8	0.5058435	0.49724034
D9	6.69865152	6.54399882

So far, we have confirmed two methods to enhance the classification accuracy of KNN algorithms: utilizing DBGBC based on the idea of DPC, and selecting the optimal value of k using data-driven neighborhood.

5.4. DBGBKNN VS DBGBKNN-3WD (RP. 3)

In the third part of the experiment, we compare the fuzziness loss of DBGBKNN and DBGBKNN-3WD. The fuzziness loss of the two algorithms after classification on nine datasets is shown in Fig. 4.

The fuzziness loss of the proposed DBGBKNN-3WD model in the actual classification process is much smaller than that of DBGBKNN. Therefore, the model guarantees the advantage of DBGBKNN without additional loss of fuzziness. Using the 3wd theory, the boundary region of this model is further subdivided when elements are added to the dataset. Therefore, the DBGBKNN-3WD reduces fuzziness loss compared to the traditional two-way decision, as demonstrated by quantitative results obtained from experiments.

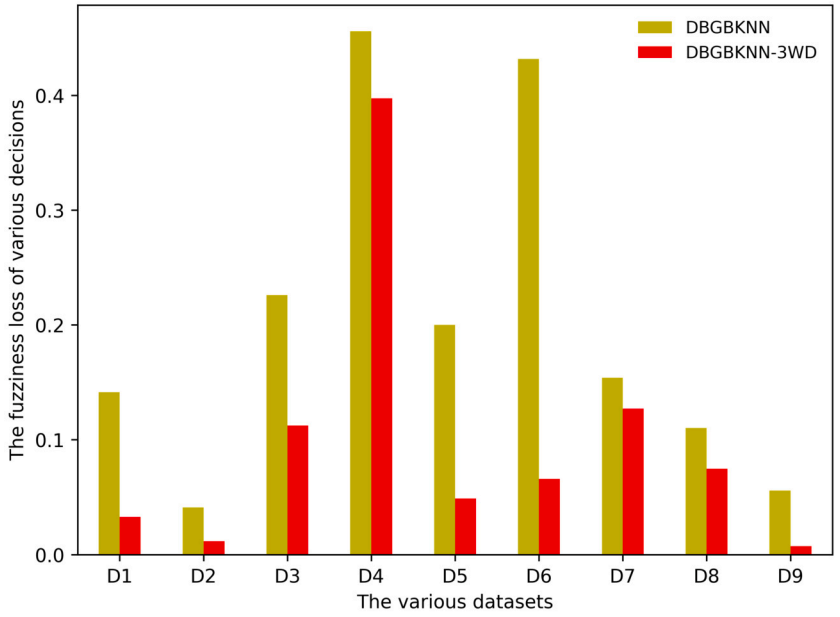

Fig. 4. The fuzziness loss of nine different datasets by various decisions.

Table 5
The accuracy of traditional KNN.

Datasets	1	3	5	7	9	11	13	15
D1	**0.951610169**	0.947387006	0.945404896	0.943163842	0.940141243	0.937288136	0.934769976	0.93309322
D2	0.949937888	0.95636646	0.959951691	0.962106625	0.963113872	0.963547274	0.96426501	**0.964267598**
D3	0.924565656	0.925565656	**0.926212121**	0.926035353	0.925925252	0.925016835	0.925085137	0.925762626
D4	0.722269737	0.752587719	0.768980263	**0.772217653**	0.771243421	0.76346674	0.754337406	0.746052632
D5	0.998540146	0.998540146	0.998540146	0.998540146	0.998540146	0.998540146	0.998540146	**0.998631387**
D6	0.636332083	0.643275619	0.646069365	0.64740772	0.648400399	0.650219346	0.651553344	**0.652434225**
D7	0.871320755	0.878018868	0.881383648	0.884245283	0.886792453	0.888427673	0.88967655	**0.890754717**
D8	**0.985361793**	0.978835836	0.976455313	0.974526476	0.973024679	0.972043881	0.971308189	0.970664055
D9	0.950997204	0.956985206	0.960572975	0.962568917	0.964045645	0.965287268	0.966342054	**0.967169868**

5.5. DBGBKNN-3WD VS other KNN algorithms (RP. 4)

Statistical Analysis. Since different values of k have different effects on traditional KNN, the accuracy of different datasets using the traditional KNN algorithm with varying values of k is initially calculated in Table 5. This ensures that the appropriate k value has been selected for the traditional KNN when performing the subsequent comparison.

Eleven KNN algorithms are included for comparison experiments. Table 6 presents four evaluation metrics, namely AC, P, Re and $F1$. Then, we perform a statistical analysis of the results, including Friedman test, Wilkerson ranksum test, and mean ranking analysis. $Win/Loss$ represents the ratio of victory and defeat after pair-to-pair comparison between DBGBKNN-3WD and other algorithms. $p-value$ reflects the difference between DBGBKNN-3WD and other algorithms. If $p-value < 0.05$, there is a significant difference between DBGBKNN-3WD and current algorithm; otherwise, there is no statistical difference between them. $Rank$ represents the average rank. The higher the rank value, the more effective the algorithm is.

As shown in Table 6, the statistical analysis of eleven algorithms, including KNN, ML-KNN, NC, FRNN-T, FRNN-G, GBKNN, GBKNN-I, GBKNN-II, GBKNN-3WD, DBGBKNN, and DBGBKNN-3WD, reveals the better performance of DBGBKNN-3WD. Among the considered metrics, DBGBKNN-3WD achieves the highest rank, indicating its overall effectiveness and highest comprehensive score. Furthermore, DBGBKNN-3WD demonstrates a significantly higher $Win/Loss$ ratio, outperforming other algorithms in 301 cases while being outperformed in only 59 cases. The $p-values$ analysis confirms the significant differences between DBGBKNN-3WD and several algorithms, underscoring its superiority. These findings highlight the advantages of DBGBKNN-3WD over the other algorithms, emphasizing its potential for solving the classification problem.

Time Comparison. As well know, the quality of a classification algorithm not only depends on the comprehensive score of classification results, but also its time consumption.

Table 7 presents the time required for classification by different algorithms on various datasets. With the increase in dataset size, the required time for classification by KNN, ML-KNN, FRNN-T and FRNN-G increases significantly, indicating that them have high time complexity when processing large datasets. In contrast, other algorithms are less affected by dataset size, where NC is almost unaffected by dataset size. However, from Table 6, the comprehensive score of NC is far lower than other algorithms.

Table 6
The statistical analysis of various algorithms.

Datasets	Metrics	KNN	ML-KNN	NC	FRNN-T	FRNN-G	GBKNN	GBKNN-I	GBKNN-II	GBKNN-3WD	DBGBKNN	DBGBKNN-3WD
D1	*AC*	0.951610169	0.939774011	0.933107345	0.884865304	0.951610169	0.89240113	0.924519774	0.9009887	0.921196669	0.936468926	0.95552029
	P	0.9534401	0.945509511	0.951853147	0.884865304	0.9534401	0.967540926	0.960737817	0.935652876	0.974307914	0.941313392	0.967790522
	Re	0.996261356	0.9907058	0.975506639	1	0.996261356	0.913417191	0.956708595	0.956638714	0.940389179	0.992487771	0.98577591
	*F*1	0.973929584	0.967362525	0.9629488	0.938825778	0.973929584	0.933477186	0.957423045	0.942597012	0.94811635	0.965730555	0.976152562
D2	*AC*	0.964285714	0.962836439	0.961428571	0.930030335	0.925652174	0.961387163	0.964244306	0.958550725	0.981219993	0.965693582	0.976168478
	P	0.972565439	0.980362392	0.966742781	0.928288529	0.921431007	0.986626323	0.984419789	0.982268499	1	0.982555009	0.993039762
	Re	0.973913043	0.96294686	0.976086956	0.952237585	0.97173913	0.954251208	0.960772947	0.954154589	0.971097235	0.965120773	0.970688738
	*F*1	0.972791683	0.971343204	0.970789068	0.939094382	0.94523132	0.969665804	0.972176326	0.967324211	0.984922626	0.973417496	0.981418005
D3	*AC*	0.92750505	0.924505051	0.918474747	0.933333329	0.924575757	0.918464646	0.920484848	0.934545454	0.994278997	0.934545454	0.994051724
	P	0.933576151	0.931640735	0.94641749	0.933333329	0.897442387	0.93146277	0.931759212	0.934631826	0.999999999	0.934631826	0.999999999
	Re	0.921591837	0.917510204	0.887265306	0.999999995	0.961714286	0.903387755	0.907387755	0.93555102	0.990890269	0.93555102	0.991106719
	*F*1	0.927221008	0.923614286	0.915268893	0.957575748	0.927845957	0.916774669	0.91912396	0.934542794	0.995338748	0.934542794	0.995452191
D4	*AC*	0.781929824	0.750625	0.600548246	0.653879359	0.659692982	0.727664474	0.726611842	0.713048246	0.755809874	0.721425439	0.761625313
	P	0.838886038	0.790132679	0.732644292	0.653879359	0.657591452	0.743897511	0.748068586	0.704449515	0.755809874	0.712501421	0.761625313
	Re	0.829493087	0.845212493	0.607706093	1	1	0.892985151	0.883358935	0.96968766	1	0.966513057	1
	*F*1	0.830427817	0.813147284	0.656212649	0.79022991	0.793409532	0.810609363	0.808610303	0.815646699	0.860018735	0.819781908	0.863276648
D5	*AC*	0.999270073	0.999270073	0.852655242	0.997815508	0.998540146	0.998540146	0.998540146	0.977382841	1	0.970808209	1
	P	0.998387097	0.998387097	0.849693441	0.99516129	0.996774193	0.996774193	0.996774193	0.96068192	1	0.967499655	1
	Re	1	1	0.813114754	1	1	1	1	0.990163934	1	0.967213115	1
	*F*1	0.999186987	0.999186987	0.830591104	0.99756097	0.998373979	0.998373979	0.998373979	0.974999305	0.999999995	0.967138365	0.999999995
D6	*AC*	0.658600392	0.678234255	0.634900692	0.617170379	0.620054387	0.654764609	0.653090542	0.64806375	0.883033787	0.649496862	0.883386296
	P	0.432432965	0.336666665	0.44550518	0.396181185	0.39895968	0.437656358	0.434910125	0.437208016	0.656666663	0.44101409	0.65833333
	Re	0.263969466	0.026799765	0.668050499	0.415419847	0.410892543	0.311397534	0.312178508	0.390986494	0.314880952	0.398637698	0.32047619
	*F*1	0.324600923	0.041933925	0.530261316	0.404098793	0.404081823	0.360135311	0.359452946	0.410207936	0.381839822	0.416039986	0.390497831
D7	*AC*	0.898301887	0.897358491	0.560377358	0.869622641	0.867169811	0.891132075	0.89245283	0.888679245	0.964349747	0.891320755	0.958006204
	P	0.900810859	0.904535954	0.508072625	0.850727771	0.845817054	0.893759162	0.894061851	0.878878638	0.976753142	0.881548733	0.97134211
	Re	0.869102578	0.863225543	0.585393398	0.860268411	0.860683261	0.859850016	0.862792965	0.872045527	0.944837687	0.875408644	0.93485569
	*F*1	0.884355074	0.88273321	0.543858602	0.855329277	0.853064526	0.876148708	0.87780069	0.87528871	0.960249724	0.87826539	0.952366824
D8	*AC*	0.985361793	0.970093281	0.812047304	0.319677098	0.985361793	0.983383776	0.944366483	0.862515678	0.931263329	0.922599355	0.939523903
	P	0.978749337	0.974788626	0.878598623	0.319677098	0.978749337	0.982023228	0.967038217	0.892048183	0.982601653	0.952781738	0.982275073
	Re	0.998574822	0.974584323	0.799049316	1	0.998574822	0.987173397	0.941805226	0.849602421	0.865105702	0.906175772	0.895725065
	*F*1	0.987423344	0.971723	0.797804881	0.462495093	0.987423344	0.984017589	0.940352586	0.860346545	0.906333607	0.92466709	0.929681492
D9	*AC*	0.972964565	0.969144438	0.861061444	0.95305403	0.957976557	0.963710982	0.963417134	0.963417835	0.980942864	0.96283041	0.979673306
	P	0.904041879	0.888677109	0.687243121	0.832823581	0.850379444	0.879021615	0.877615761	0.865293658	0.931318835	0.863484799	0.930175993
	Re	0.922916159	0.91649027	0.963385846	0.900221919	0.902192362	0.898739209	0.898244159	0.925871824	0.940944979	0.926864361	0.938330272
	*F*1	0.912155524	0.900511311	0.768362196	0.859546799	0.871146297	0.885671363	0.884778753	0.89002876	0.934925287	0.889179378	0.933026731
Statistics	*Win/Loss*	29/7	31/5	32/4	29/7	27/9	33/3	33/3	34/2	21/15	32/4	**301**/59
	$p-value$	0.140115799	0.05003807	**1.97E-05**	**0.01775386**	0.10728713	**0.037203524**	**0.018026136**	**0.002107815**	0.901406182	**0.00559663**	-
	Rank	7.583333**(3)**	6.25	3.611111	3.972222	5.361111	5.097222	5.319444	4.694444	8.833333**(2)**	6	**9.277778(1)**

Table 7
The required time of various algorithms.

Datasets	KNN	ML-KNN	NC	FRNN-T	FRNN-G	GBKNN	GBKNN-I	GBKNN-II	GBKNN-3WD	DBGBKNN	DBGBKNN-3WD
D1	0.4137908	1.45828929	0.00287676	0.24015344	0.24019499	0.04426174	0.05687038	0.04507522	0.0450258	0.05810616	0.0584077
D2	0.57012691	1.97220619	0.00274521	0.35937604	0.34724257	0.02650604	0.06483773	0.02788872	0.02748004	0.06580449	0.06583999
D3	1.15815918	3.97770792	0.0027762	0.66695051	0.66210109	0.04839858	0.05019313	0.05017558	0.05039779	0.05250887	0.05284704
D4	1.11194303	3.77993105	0.00291289	0.63315068	0.62276879	0.21575751	0.21516508	0.21246095	0.21213527	0.22061626	0.21830661
D5	2.22822806	7.63954103	0.00287477	1.29076884	1.29366515	0.0676181	0.11637658	0.06958162	0.06984887	0.11921921	0.1185345
D6	21.02310103	71.25166182	0.00389255	12.13694283	12.20975923	3.29948898	3.50095659	3.33018495	3.33016917	3.53015668	3.5360387
D7	34.67396715	119.9659204	0.00479799	19.38198602	19.61077245	3.28217726	5.50588911	3.31035479	3.30475013	5.40372487	5.43626809
D8	81.13092173	274.4331901	0.00626588	45.89641048	45.81044944	0.5058435	9.36806221	0.49724034	0.51140343	9.40523808	9.40388384
D9	225.2088671	771.9289203	0.0201305	128.4236102	128.5592262	6.69865152	8.10988375	6.54399882	6.58448443	7.99999576	8.09387493
$p-value$	**0.070265673**	**0.030510209**	**0.000348575**	**0.070265673**	**0.070265673**	0.3098795496	0.8252828980	0.3098795496	0.3098795496	0.89462581	-
Rank	10**(10)**	11**(11)**	1**(1)**	8.444444**(8)**	8.555556**(9)**	2.666667**(2)**	5.222222**(5)**	3.222222**(3)**	3.333333**(4)**	6.111111**(6)**	6.444444**(7)**

In addition, we conduct a statistical analysis of the Table 7. On the one hand, DBGBKNN-3WD is significantly different from KNN, ML-KNN, FRNN-T, FRNN-G and NC when the $p-value$ threshold is 0.1. On the other hand, from the perspective of $Rank$, the time consumed by DBGBKNN-3WD ranks 7th among the eleven algorithms. That is to say, the required time of DBGBKNN-3WD corresponds to that of other granular ball-based algorithms. Both of them are far larger than NC in the first place and far smaller than KNN and ML-KNN in the last place. In general, DBGBKNN-3WD provides a classification solution with relatively low time complexity.

In conclusion, the DBGBKNN-3WD algorithm improves the accuracy of GBKNN, and outperforms other KNN algorithms in terms of comprehensive score, as measured by specific evaluation criteria in experiments.

6. Conclusions

This research presents the DBGBKNN-3WD by introducing the 3WD theory, which enhances the GBKNN method through data-driven optimizations of neighborhood size selection and the granular ball generation process. By refining granular ball generation and selecting optimal local k values, the DBGBKNN approach increases the classification accuracy, outperforming the existing methods. The three-way decision model further reduces uncertainty compared to traditional binary choices, providing a more robust classification solution. Experimental results demonstrate that the proposed method has significant improvements in classification accuracy and robustness, while maintaining reasonable time complexity. Therefore, our approach provides a solid foundation for future work in developing more robust, efficient, and accurate KNN classifiers.

Future researches will focus on expanding the optimization objectives for threshold determination, incorporating additional criteria like misclassification cost and class separation distance in a multi-objective framework. An optimal granular space selection mechanism will be designed by considering the user requirements for decision precision and computational expense, respectively. Additionally, an incremental learning based on 3DBGBKNN-3WD will be designed for online feature selection by considering the updating mechanism of granular-balls.

CRediT authorship contribution statement

Jie Yang: Writing – review & editing, Writing – original draft, Validation, Methodology, Formal analysis, Conceptualization. **Juncheng Kuang:** Writing – original draft, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Guoyin Wang:** Writing – review & editing, Supervision. **Qinghua Zhang:** Writing – review & editing, Supervision. **Yanmin Liu:** Writing – review & editing, Supervision. **Qun Liu:** Writing – review & editing, Supervision. **Deyou Xia:** Writing – review & editing. **Shuai Li:** Writing – review & editing. **Xiaoqi Wang:** Writing – review & editing. **Di Wu:** Writing – review & editing, Validation, Supervision, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgement

This work was supported by the National Science Foundation of China (Grant number 62066049, Grant number 62221005, Grant number 62176070, Grant number 62366033), the Guizhou Provincial Department of Education Colleges and Universities Science and Technology Innovation Team (QJJ [2023] 084), Excellent Young Scientific and Technological Talents Foundation of Guizhou Province (QKH-platform talent (2021) 5627), Science and Technology Top Talent Project of Guizhou Education Department (QJJ2022 [088]), Science and Technology Project of Zunyi (ZSKRPT [2023] 3).

References

[1] W. Pedrycz, X. Wang, Optimal granularity of machine learning models: a perspective of granular computing, IEEE Transactions on Fuzzy Systems 32 (4) (2024) 2176–2186.
[2] W. Pedrycz, Granular data compression and representation, IEEE Trans. Fuzzy Syst. 31 (5) (2023) 1497–1505.
[3] Y.Y. Yao, J.L. Yang, Granular fuzzy sets and three-way approximations of fuzzy sets, Int. J. Approx. Reason. 161 (2023) 109003.
[4] J.T. Yao, A.V. Vasilakos, W. Pedrycz, Granular computing: perspectives and challenges, IEEE Trans. Cybern. 43 (6) (2013) 1977–1989.
[5] W. Pedrycz, R. Al-Hmouz, A. Morfeq, A. Balamash, The design of free structure granular mappings: the use of the principle of justifiable granularity, IEEE Trans. Cybern. 43 (6) (2013) 2105–2113.
[6] G. Wang, J. Yang, J. Xu, Granular computing: from granularity optimization to multi-granularity joint problem solving, Granul. Comput. 2 (3) (2017) 105–120.
[7] Y.Y. Yao, Three-way decision and granular computing, Int. J. Approx. Reason. 103 (2018) 107–123.
[8] T.H. Ouyang, W. Pedrycz, O.F. Reyes-Galaviz, N.J. Pizzi, Granular description of data structures: a two-phase design, IEEE Trans. Cybern. 51 (4) (2021) 1902–1912.
[9] T. Cover, P. Hart, Nearest neighbor pattern classification, IEEE Trans. Inf. Theory 13 (1) (1967) 21–27.

[10] M. Hu, E.C. Tsang, Y.T. Guo, D.G. Chen, W.H. Xu, Attribute reduction based on overlap degree and k-nearest-neighbor rough sets in decision information systems, Inf. Sci. 584 (2022) 301–324.
[11] Y.X. Dong, X.J. Ma, T.L. Fu, Electrical load forecasting: a deep learning approach based on k-nearest neighbors, Appl. Soft Comput. 99 (2021) 106900.
[12] J.W. Yang, X. Tan, S. Rahardja, Outlier detection: how to select k for k-nearest-neighbors-based outlier detectors, Pattern Recognit. Lett. 174 (2023) 112–117.
[13] M.L. Zhang, Z.H. Zhou, Ml-knn: a lazy learning approach to multi-label learning, Pattern Recognit. 40 (7) (2007) 2038–2048.
[14] C. Li, S.F. Ding, X. Xu, H.W. Hou, L. Ding, Fast density peaks clustering algorithm based on improved mutual k-nearest-neighbor and sub-cluster merging, Inf. Sci. 647 (2023) 119470.
[15] J.P. Gou, L.Y. Sun, L. Du, H.X. Ma, T.S. Xiong, W.H. Ou, Y.Z. Zhan, A representation coefficient-based k-nearest centroid neighbor classifier, Expert Syst. Appl. 194 (2022) 116529.
[16] X.J. Duan, Y. Ma, Y.Q. Zhou, H. Huang, B. Wang, A novel cluster validity index based on augmented non-shared nearest neighbors, Expert Syst. Appl. 223 (2023) 119784.
[17] S.Y. Xia, Y.S. Liu, X. Ding, G.Y. Wang, H. Yu, Y.G. Luo, Granular ball computing classifiers for efficient, scalable and robust learning, Inf. Sci. 483 (2019) 136–152.
[18] S.Y. Xia, X.C. Dai, G.Y. Wang, X.B. Gao, E. Giem, An efficient and adaptive granular-ball generation method in classification problem, IEEE Trans. Neural Netw. Learn. Syst. 35 (4) (2022) 5319–5331.
[19] S.Y. Xia, H. Zhang, W.H. Li, G.Y. Wang, E. Giem, Z.Z. Chen, Gbnrs: a novel rough set algorithm for fast adaptive attribute reduction in classification, IEEE Trans. Knowl. Data Eng. 34 (3) (2022) 1231–1242.
[20] S.Y. Xia, D.W. Peng, D.Y. Meng, C.Q. Zhang, G.Y. Wang, E. Giem, W. Wei, A fast adaptive k-means with no bounds, IEEE Trans. Pattern Anal. Mach. Intell. 44 (1) (2020) 87–99.
[21] S.Y. Xia, S.Y. Zheng, G.Y. Wang, X.B. Gao, B.G. Wang, Granular ball sampling for noisy label classification or imbalanced classification, IEEE Trans. Neural Netw. Learn. Syst. 34 (4) (2023) 2144–2155.
[22] S.Y. Xia, Y. Zheng, G.Y. Wang, P. He, H. Li, Z.Z. Chen, Random space division sampling for label-noisy classification or imbalanced classification, IEEE Trans. Cybern. 52 (10) (2022) 10444–10457.
[23] Q.H. Zhang, C.Y. Wu, S.Y. Xia, F. Zhao, M. Gao, Y.L. Cheng, G.Y. Wang, Incremental learning based on granular ball rough sets for classification in dynamic mixed-type decision system, IEEE Trans. Knowl. Data Eng. 35 (9) (2023) 9319–9332.
[24] Y. Chen, P.X. Wang, X.B. Yang, J.S. Mi, D. Liu, Granular ball guided selector for attribute reduction, Knowl.-Based Syst. 229 (2021) 107326.
[25] X.L. Peng, P. Wang, S.Y. Xia, C. Wang, W.Q. Chen, Vpgb: a granular-ball based model for attribute reduction and classification with label noise, Inf. Sci. 611 (2022) 504–521.
[26] A. Rodriguez, A. Laio, Clustering by fast search and find of density peaks, Science 344 (6191) (2014) 1492–1496.
[27] Y.Z. Wang, D. Wang, Y. Zhou, X.F. Zhang, C. Quek, Vdpc: variational density peak clustering algorithm, Inf. Sci. 621 (2023) 627–651.
[28] J. Zhao, G. Wang, J. Pan, F.T. Huai, I. Lee, Density peaks clustering algorithm based on fuzzy and weighted shared neighbor for uneven density datasets, Pattern Recognit. 139 (2023) 109406.
[29] Y.Y. Yao, The dao of three-way decision and three-world thinking, Int. J. Approx. Reason. 162 (2023) 109032.
[30] Y.Y. Yao, J.L. Yang, Granular fuzzy sets and three-way approximations of fuzzy sets, Int. J. Approx. Reason. 161 (2023) 109003.
[31] J. Deng, J.M. Zhan, Z. Xu, E. Herrera-Viedma, Regret-theoretic multiattribute decision-making model using three-way framework in multiscale information systems, IEEE Trans. Cybern. 53 (6) (2023) 3988–4001.
[32] W.J. Wang, J.M. Zhan, E. Herrera-Viedma, A three-way decision approach with a probability dominance relation based on prospect theory for incomplete information systems, Inf. Sci. 611 (2022) 199–224.
[33] T.B. Li, J.S. Qiao, W.P. Ding, Three-way conflict analysis and resolution based on q-rung orthopair fuzzy information, Inf. Sci. 638 (2023) 118959.
[34] M.J. Du, J.Q. Zhao, J.R. Sun, Y.Q. Dong, M3w: multistep three-way clustering, IEEE Trans. Neural Netw. Learn. Syst. 35 (4) (2022) 1–14.
[35] J.M. Zhan, J.J. Wang, W.P. Ding, Y.Y. Yao, Three-way behavioral decision making with hesitant fuzzy information systems: survey and challenges, IEEE/CAA J. Autom. Sin. 10 (2) (2022) 330–350.
[36] R.T. Zhang, X.L. Ma, J.M. Zhan, Y.Y. Yao, 3wc-d: a feature distribution-based adaptive three-way clustering method, Appl. Intell. 53 (12) (2023) 15561–15579.
[37] R.R. Zhao, L.N. Ma, S.G. Li, M.X. Luo, A multi-criteria three-way decision making method in a picture fuzzy probabilistic decision system, Cogn. Comput. 14 (6) (2022) 1924–1941.
[38] T.X. Wang, B. Huang, H.X. Li, D. Liu, H. Yu, Three-way decision for probabilistic linguistic conflict analysis via compounded risk preference, Inf. Sci. 631 (2023) 65–90.
[39] Q.H. Hu, D.R. Yu, J.F. Liu, C.X. Wu, Neighborhood rough set based heterogeneous feature subset selection, Inf. Sci. 178 (18) (2008) 3577–3594.
[40] D. Zhang, P. Zhu, Variable radius neighborhood rough sets and attribute reduction, Int. J. Approx. Reason. 150 (2022) 98–121.
[41] H.Y. Zhang, Q.Q. Sun, K.Z. Dong, Information-theoretic partially labeled heterogeneous feature selection based on neighborhood rough sets, Int. J. Approx. Reason. 154 (2023) 200–217.
[42] J.J. Chen, S.H. Yu, W.J. Wei, Y. Ma, Matrix-based method for solving decision domains of neighbourhood multigranulation decision-theoretic rough sets, CAAI Trans. Intell. Technol. 7 (2022) 313–327.
[43] Q.H. Zhang, P. Zhang, G.Y. Wang, Research on approximation set of rough set based on fuzzy similarity, J. Intell. Fuzzy Syst. 32 (2017) 2549–2562.
[44] Q.H. Zhang, Y. Xiao, G.Y. Wang, A new method for measuring fuzziness of vague set or intuitionistic fuzzy set, J. Intell. Fuzzy Syst. 25 (2) (2013) 505–515.
[45] D. Dai, P. Rigollet, T. Zhang, Deviation optimal learning using greedy Q-aggregation, Ann. Stat. 40 (3) (2012) 1878–1905.
[46] Y. Ma, R. Huang, M. Yan, G.Q. Li, T. Wang, Attention-based local mean k-nearest centroid neighbor classifier, Expert Syst. Appl. 201 (2022) 117159.
[47] R. Jensen, C. Cornelis, Fuzzy-rough nearest neighbour classification and prediction, Theor. Comput. Sci. 412 (42) (2011) 5871–5884.